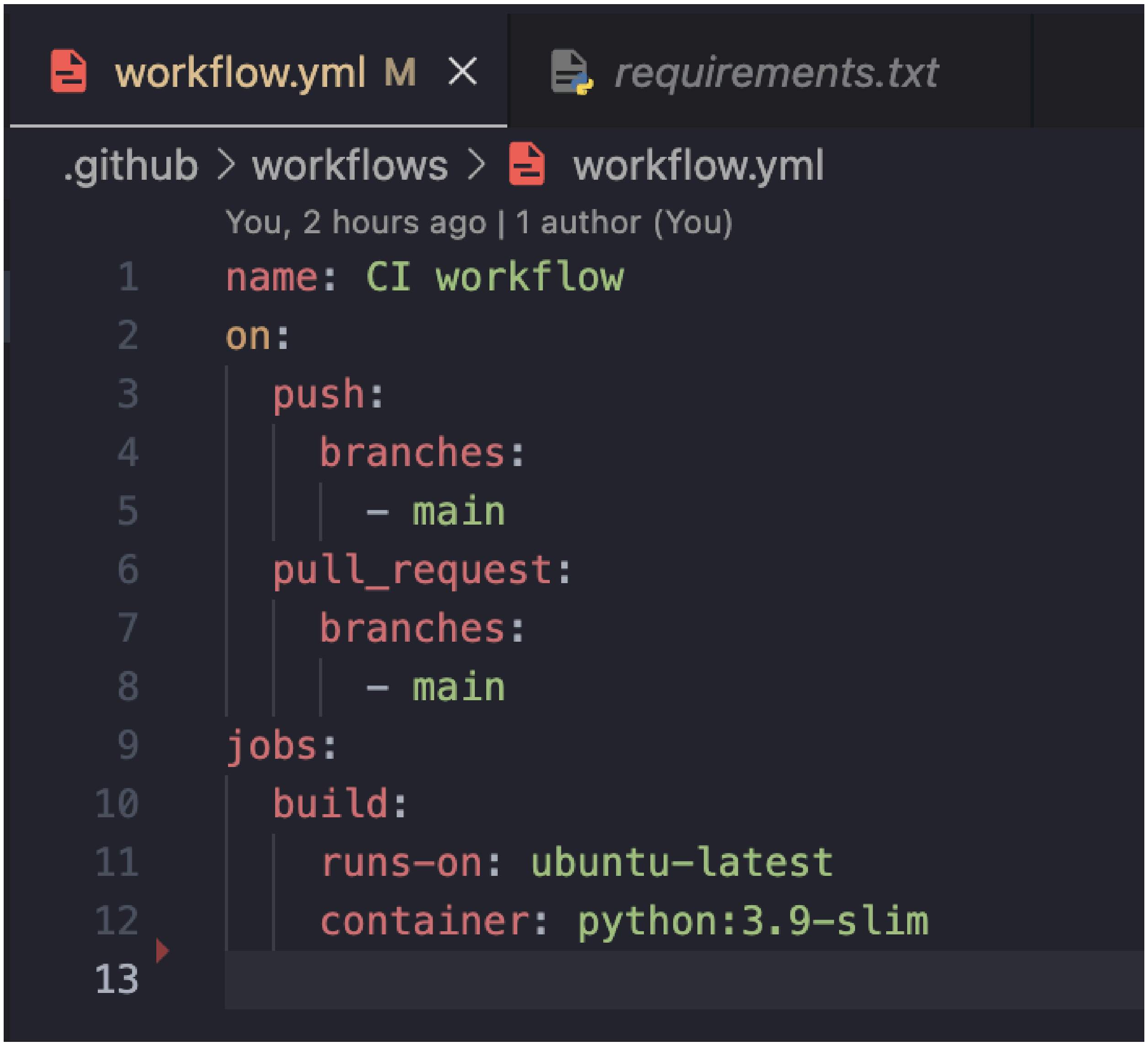
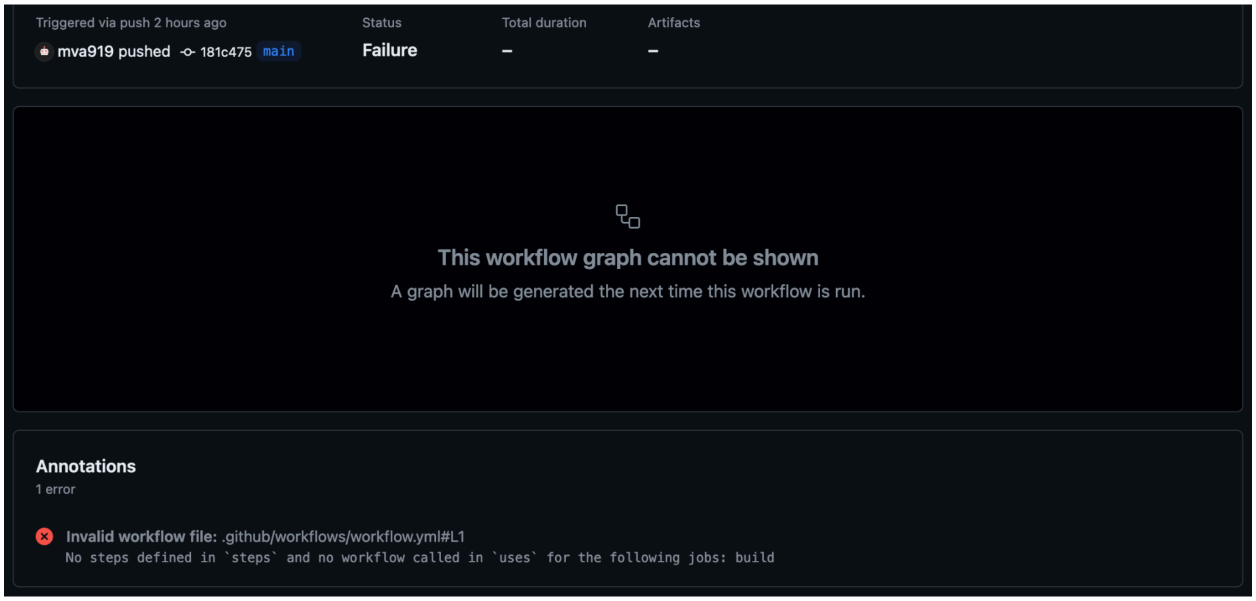
Marcos Villanueva Abreu
CS472 - CI
Clone Link: https://github.com/mva919/tdd.git

## Task 1:


workflow.yml M
requirements.txt
.github > workflows > workflow.yml
You, 2 hours ago | 1 author (You)

```yaml
name: CI workflow
on:
  push:
    branches:
      - main
  pull_request:
    branches:
      - main
jobs:
  build:
    runs-on: ubuntu-latest
    container: python:3.9-slim

```

Figure 1: Workflow code snippet for Task 1



Triggered via push 2 hours ago

mva919 pushed 181c475 main

Status

Failure

Total duration

–

Artifacts

–

This workflow graph cannot be shown

A graph will be generated the next time this workflow is run.

Annotations

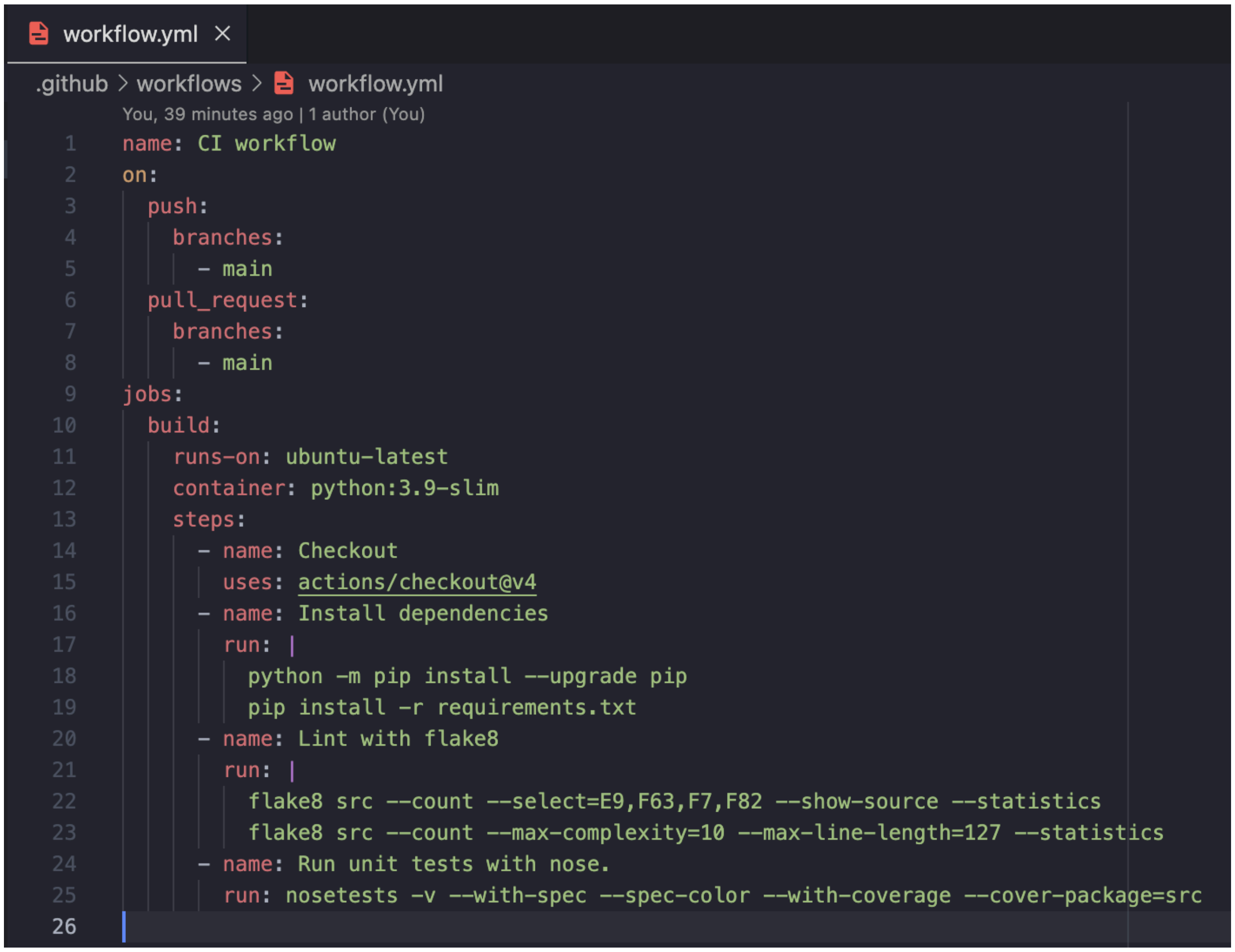
1 error

**Invalid workflow file:** .github/workflows/workflow.yml#L1

```
No steps defined in `steps` and no workflow called in `uses` for the following jobs: build
```

Figure 2: Github Actions build failure for Figure 1 code snippet

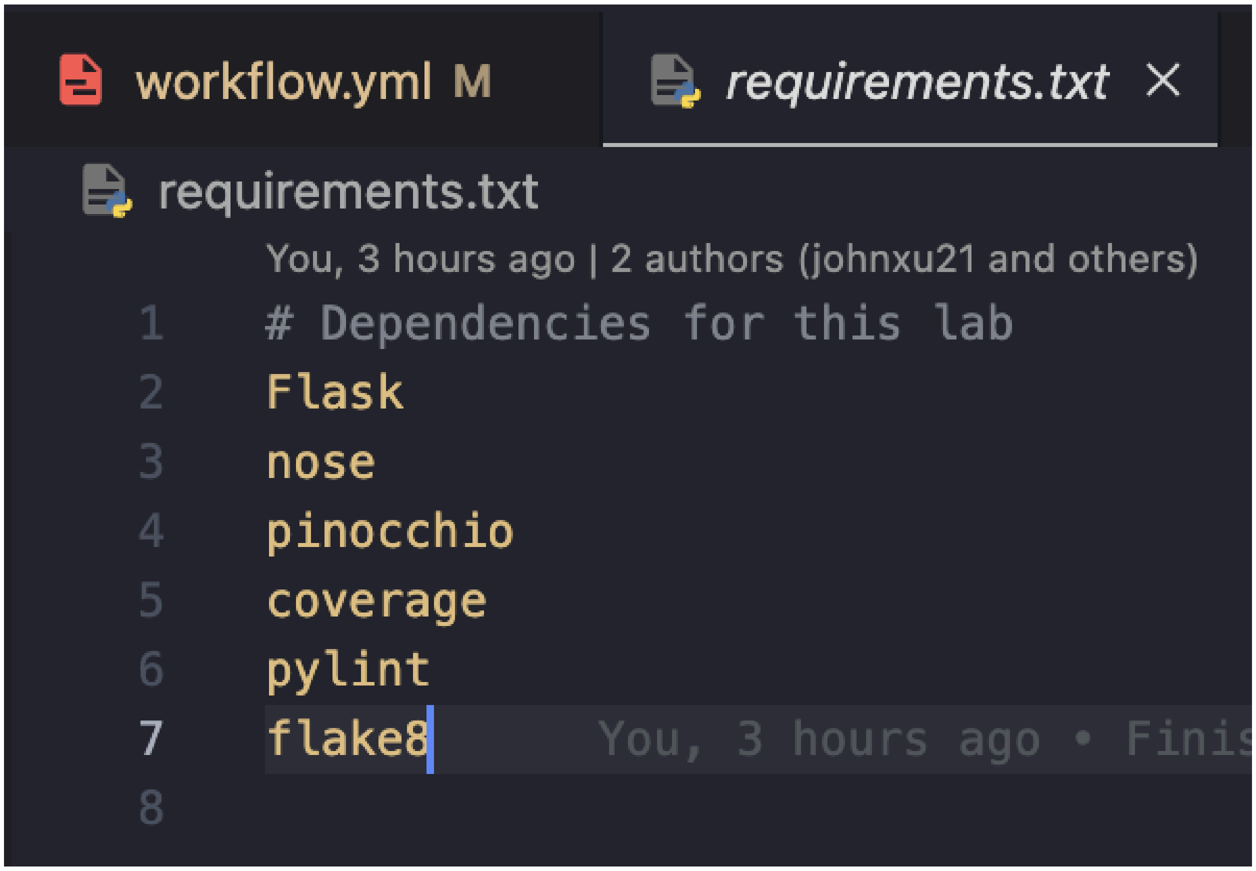
## Task 2:


workflow.yml
.github > workflows > workflow.yml
You, 39 minutes ago | 1 author (You)

```yaml
name: CI workflow
on:
  push:
    branches:
      - main
  pull_request:
    branches:
      - main
jobs:
  build:
    runs-on: ubuntu-latest
    container: python:3.9-slim
    steps:
      - name: Checkout
        uses: actions/checkout@v4
      - name: Install dependencies
        run: |
          python -m pip install --upgrade pip
          pip install -r requirements.txt
      - name: Lint with flake8
        run: |
          flake8 src --count --select=E9,F63,F7,F82 --show-source --statistics
          flake8 src --count --max-complexity=10 --max-line-length=127 --statistics
      - name: Run unit tests with nose.
        run: nosetests -v --with-spec --spec-color --with-coverage --cover-package=src

```

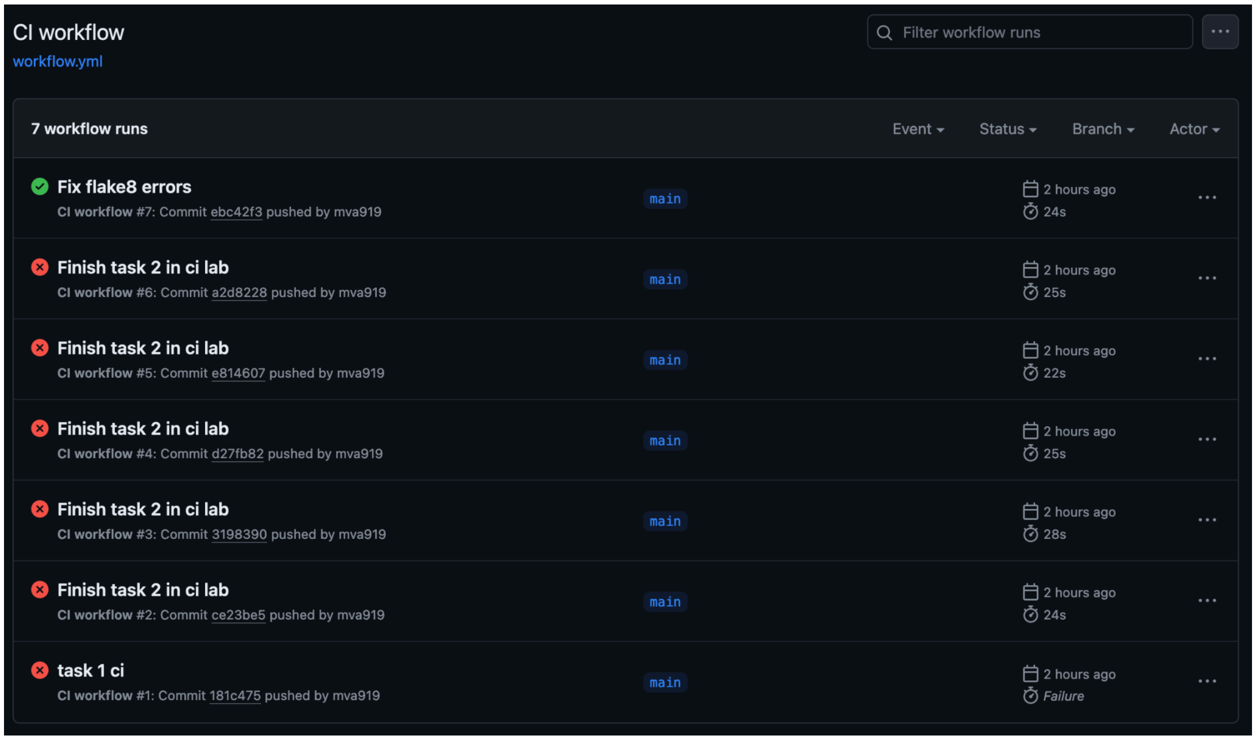
Figure 3: Completed workflow code snippet for Task 2

*Note: Used actions/checkout@v4 because I had an issue with node versions using v3.*



```
requirements.txt
You, 3 hours ago | 2 authors (johnxu21 and others)
# Dependencies for this lab
Flask
nose
pinocchio
coverage
pylint
flake8        You, 3 hours ago • Finis
```

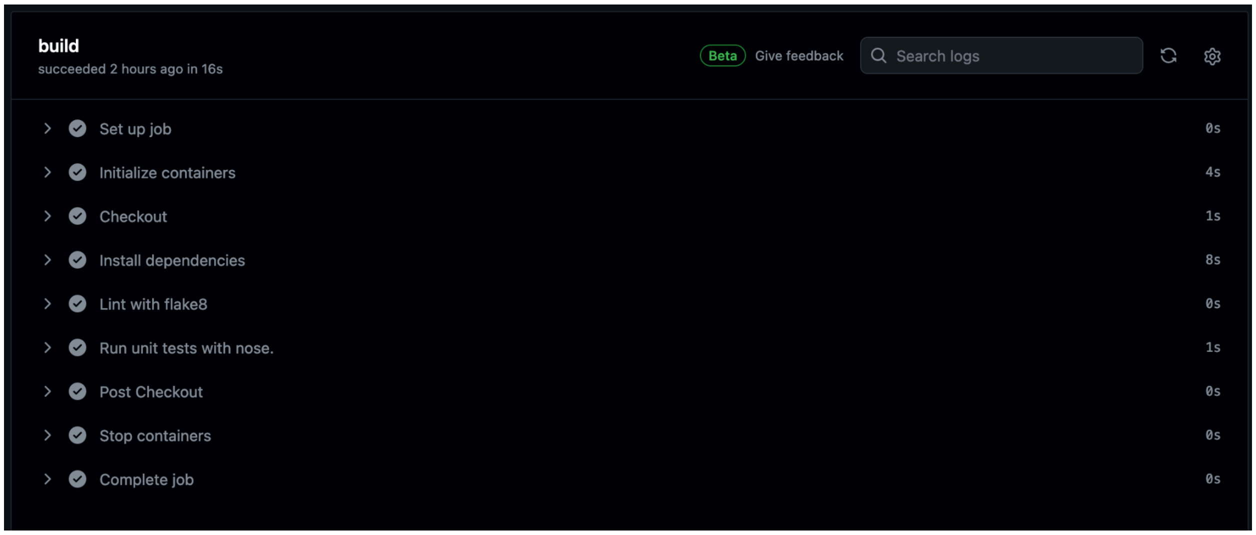
Figure 4: Code snippet for requirements.txt



7 workflow runs

| | | | |
|---|---|---|---|
| ✓ Fix flake8 errors<br>CI workflow #7: Commit ebc42f3 pushed by mva919 | main | 2 hours ago | 24s |
| ✗ Finish task 2 in ci lab<br>CI workflow #6: Commit a2d8228 pushed by mva919 | main | 2 hours ago | 25s |
| ✗ Finish task 2 in ci lab<br>CI workflow #5: Commit e814607 pushed by mva919 | main | 2 hours ago | 22s |
| ✗ Finish task 2 in ci lab<br>CI workflow #4: Commit d27fb82 pushed by mva919 | main | 2 hours ago | 25s |
| ✗ Finish task 2 in ci lab<br>CI workflow #3: Commit 3198390 pushed by mva919 | main | 2 hours ago | 28s |
| ✗ Finish task 2 in ci lab<br>CI workflow #2: Commit ce23be5 pushed by mva919 | main | 2 hours ago | 24s |
| ✗ task 1 ci<br>CI workflow #1: Commit 181c475 pushed by mva919 | main | 2 hours ago | Failure |

Figure 5: CI Workflow run history

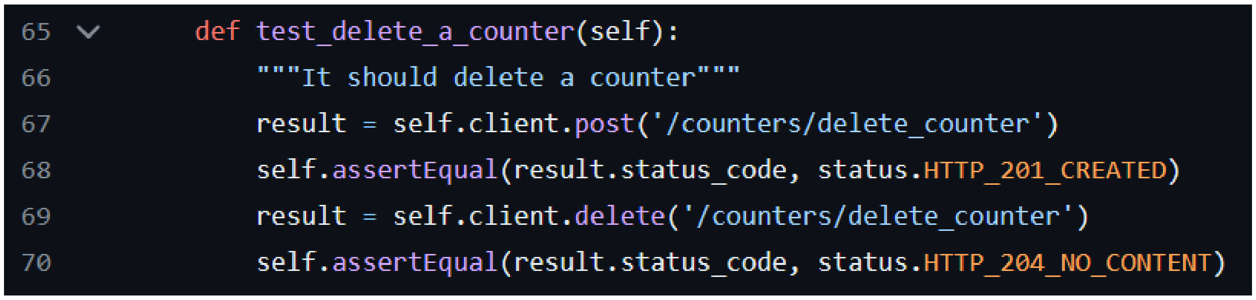


```
build
succeeded 2 hours ago in 16s

Set up job                      0s
Initialize containers           4s
Checkout                        1s
Install dependencies            8s
Lint with flake8                0s
Run unit tests with nose.       1s
Post Checkout                   0s
Stop containers                 0s
Complete job                    0s
```

Figure 6: Successful CI Workflow build

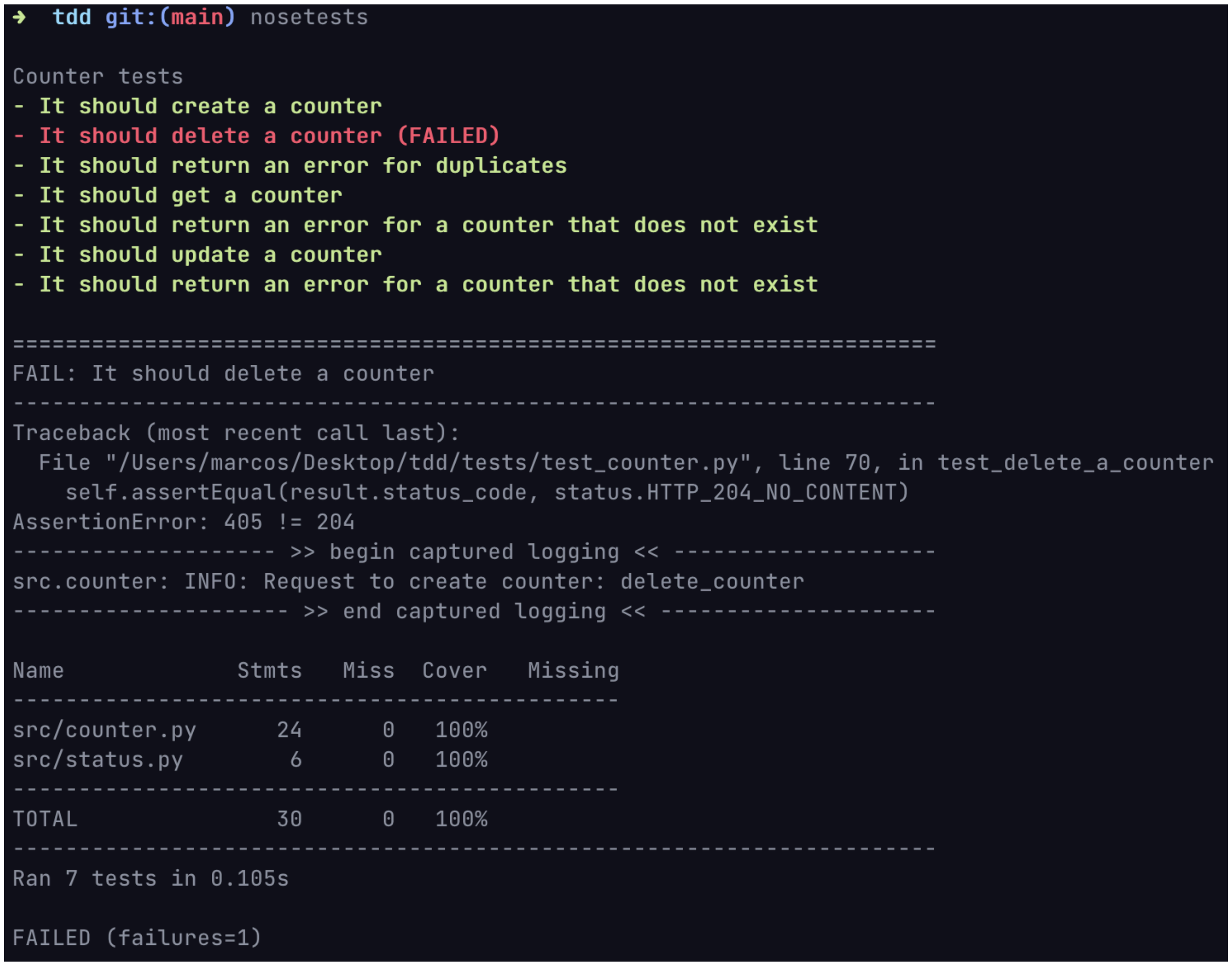
## Red Phase:

To test if a counter has been properly deleted I first want to create that counter by using a post request and asserting that the request returned successfully with a 201 created status code. Afterwards, I want to make a delete request to the counter I just created and I want to verify that the request returned a 204 code meaning it was successful in deleting said counter.



```python
    def test_delete_a_counter(self):
        """It should delete a counter"""
        result = self.client.post('/counters/delete_counter')
        self.assertEqual(result.status_code, status.HTTP_201_CREATED)
        result = self.client.delete('/counters/delete_counter')
        self.assertEqual(result.status_code, status.HTTP_204_NO_CONTENT)
```

Figure 7: Code Snippet for Red Phase

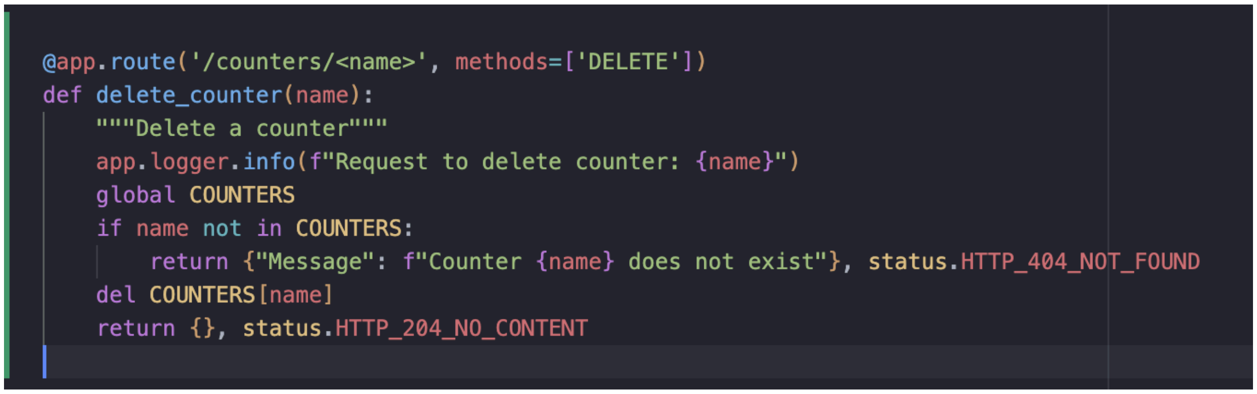


```
➜  tdd git:(main) nosetests

Counter tests
- It should create a counter
- It should delete a counter (FAILED)
- It should return an error for duplicates
- It should get a counter
- It should return an error for a counter that does not exist
- It should update a counter
- It should return an error for a counter that does not exist

======================================================================
FAIL: It should delete a counter
----------------------------------------------------------------------
Traceback (most recent call last):
  File "/Users/marcos/Desktop/tdd/tests/test_counter.py", line 70, in test_delete_a_counter
    self.assertEqual(result.status_code, status.HTTP_204_NO_CONTENT)
AssertionError: 405 != 204
-------------------- >> begin captured logging << --------------------
src.counter: INFO: Request to create counter: delete_counter
--------------------- >> end captured logging << ---------------------

Name             Stmts   Miss  Cover   Missing
----------------------------------------------
src/counter.py      24      0   100%
src/status.py        6      0   100%
----------------------------------------------
TOTAL               30      0   100%
----------------------------------------------------------------------
Ran 7 tests in 0.105s

FAILED (failures=1)
```

Figure 8: Red phase results for delete counter

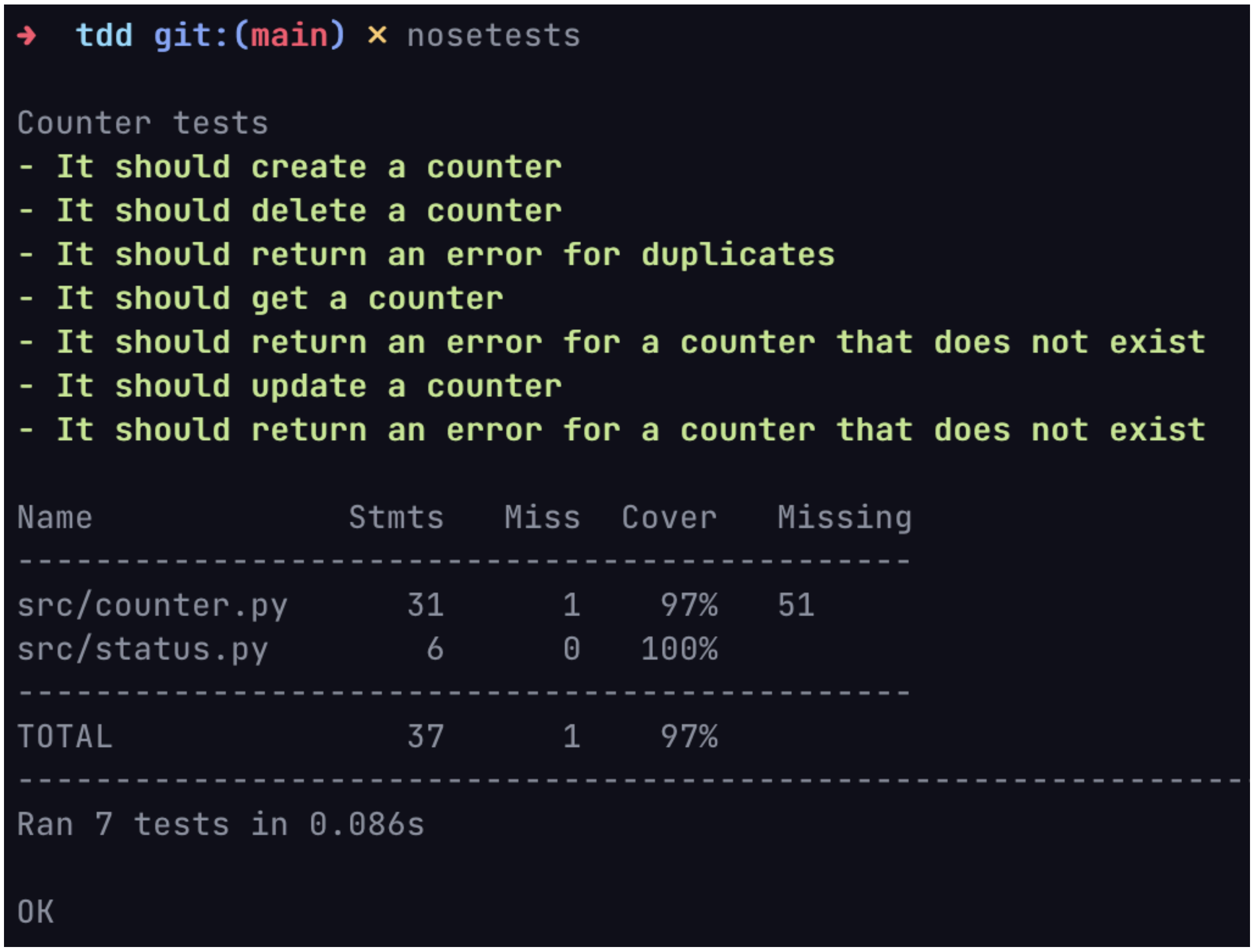
## Green Phase:

To implement a delete method for the counter route we need to first check if the counter is present in the COUNTERS dictionary, if it isn't we throw a 404 not found status code. Otherwise, we delete the counter object from the COUNTERS dictionary and return 204 no content status code to let the client know the request was successful.



```python
@app.route('/counters/<name>', methods=['DELETE'])
def delete_counter(name):
    """Delete a counter"""
    app.logger.info(f"Request to delete counter: {name}")
    global COUNTERS
    if name not in COUNTERS:
        return {"Message": f"Counter {name} does not exist"}, status.HTTP_404_NOT_FOUND
    del COUNTERS[name]
    return {}, status.HTTP_204_NO_CONTENT
```

Figure 9: Code snippet for Green Phase

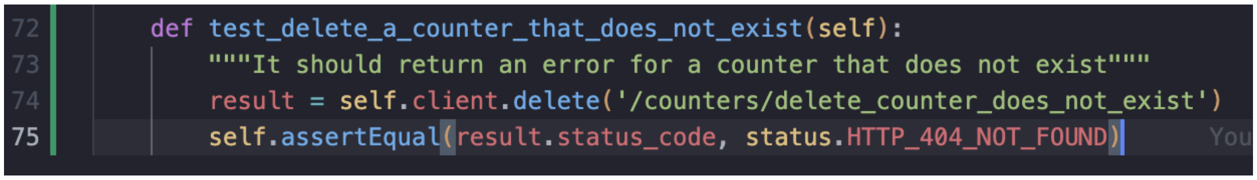


```
➜  tdd git:(main) ✗ nosetests

Counter tests
- It should create a counter
- It should delete a counter
- It should return an error for duplicates
- It should get a counter
- It should return an error for a counter that does not exist
- It should update a counter
- It should return an error for a counter that does not exist

Name                 Stmts   Miss  Cover   Missing
----------------------------------------------
src/counter.py          31      1    97%   51
src/status.py            6      0   100%
----------------------------------------------
TOTAL                   37      1    97%
-----------------------------------------------------------------
Ran 7 tests in 0.086s

OK
```

Figure 10: Results for Green Phase

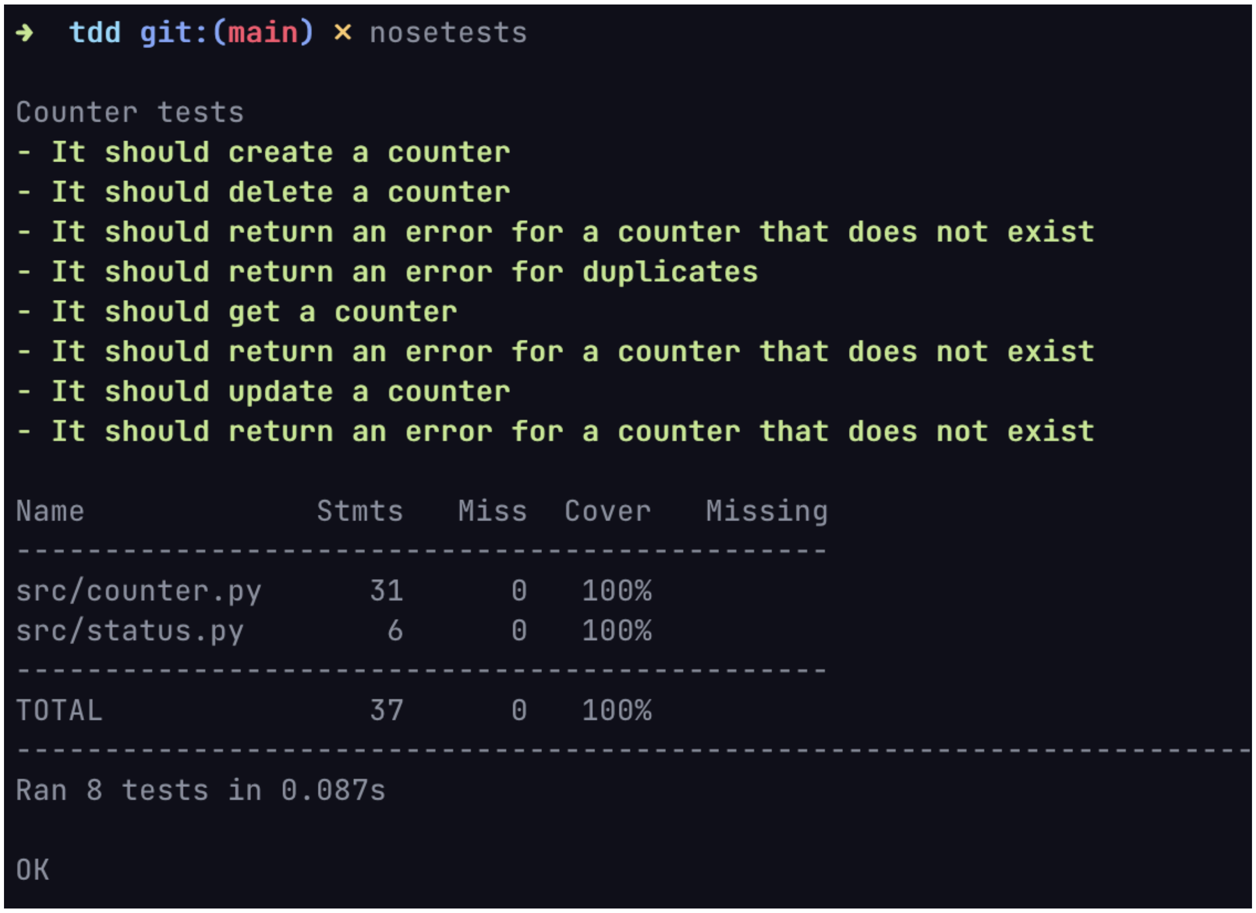
## Refactor Phase:

Since the test case I wrote did not cover 100% of the new delete method, I will add a new test that checks if the delete method handles requests of counters that do not exist in the COUNTERS dictionary; it should throw a 404 status code when this happens.



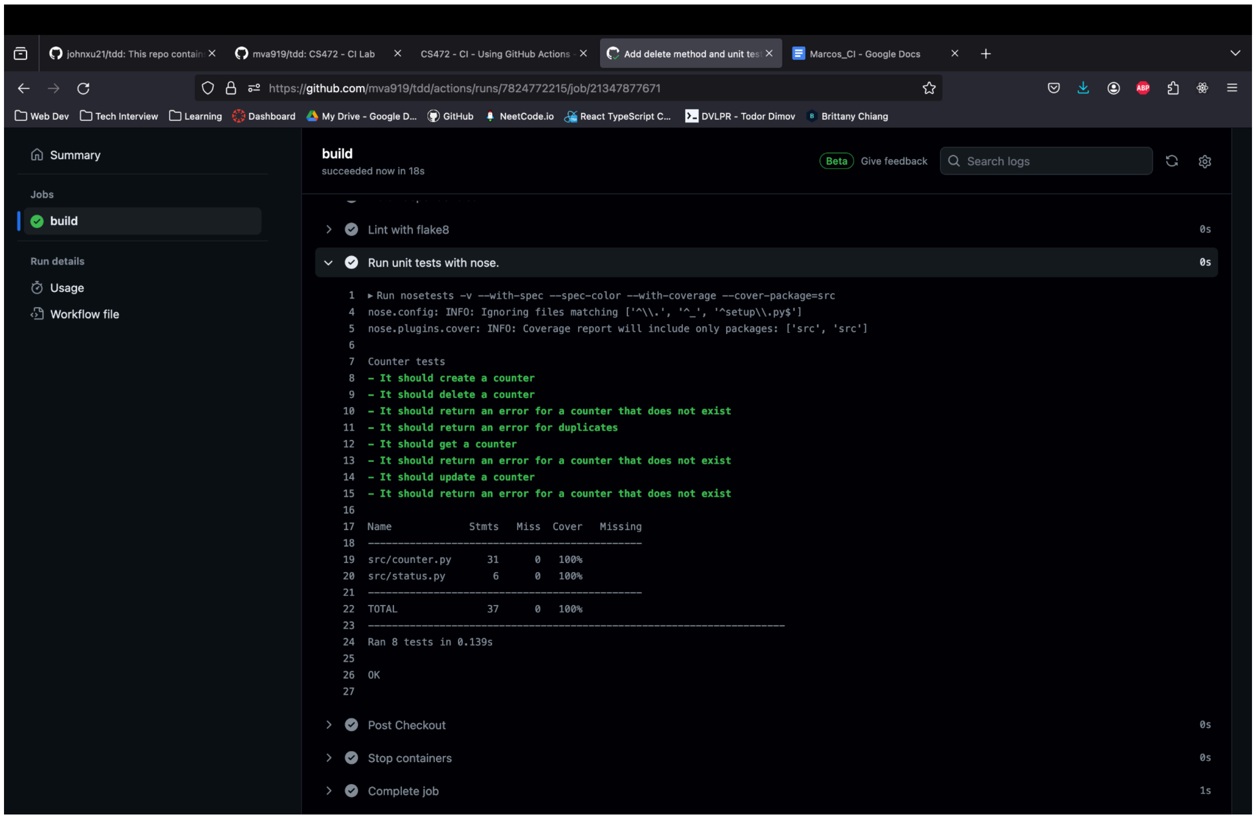
```
    def test_delete_a_counter_that_does_not_exist(self):
        """It should return an error for a counter that does not exist"""
        result = self.client.delete('/counters/delete_counter_does_not_exist')
        self.assertEqual(result.status_code, status.HTTP_404_NOT_FOUND)      You
```

Figure 11: Code Snippet for Refactor Phase



```
➜  tdd git:(main) ✗ nosetests

Counter tests
- It should create a counter
- It should delete a counter
- It should return an error for a counter that does not exist
- It should return an error for duplicates
- It should get a counter
- It should return an error for a counter that does not exist
- It should update a counter
- It should return an error for a counter that does not exist

Name                Stmts   Miss  Cover   Missing
----------------------------------------------
src/counter.py         31      0   100%
src/status.py           6      0   100%
----------------------------------------------
TOTAL                  37      0   100%
----------------------------------------------------------------------
Ran 8 tests in 0.087s

OK
```

Figure 12: Result for Refactor Phase



```
▸ Run nosetests -v --with-spec --spec-color --with-coverage --cover-package=src
nose.config: INFO: Ignoring files matching ['^\\.', '^_', '^setup\\.py$']
nose.plugins.cover: INFO: Coverage report will include only packages: ['src', 'src']

Counter tests
- It should create a counter
- It should delete a counter
- It should return an error for a counter that does not exist
- It should return an error for duplicates
- It should get a counter
- It should return an error for a counter that does not exist
- It should update a counter
- It should return an error for a counter that does not exist

Name              Stmts   Miss  Cover   Missing
-----------------------------------------------
src/counter.py       31      0   100%
src/status.py         6      0   100%
-----------------------------------------------
TOTAL                37      0   100%
----------------------------------------------------------------------
Ran 8 tests in 0.139s

OK
```

Figure 13: Full test coverage in CI Workflow build